МІНІСТЕРСТВО ОСВІТИ І НАУКИ,
МОЛОДІ ТА СПОРТУ УКРАЇНИ

Національний технічний університет
«Харківський політехнічний інститут»

МЕТОДИЧНІ ВКАЗІВКИ
до самостійної роботи
«Найпростіші введення та виведення у візуальному середовищі Delphi»

з курсу «Програмування»
для студентів напряму 6.040302 – Інформатика
(спеціалізація «Соціальна інформатика»)

Затверджено редакційно-видавничою
радою університету,
протокол № 2 від 01.12.10.

Харків НТУ «ХПІ» 2011

Методичні вказівки до самостійної роботи «Найпростіші введення та виведення у візуальному середовищі Delphi» з курсу «Програмування» для студентів напряму 6.040302 – Інформатика (спеціалізація «Соціальна інформатика») / Уклад. М. І. Безменов. – Х. : НТУ «ХПІ», 2011. – 16 с.

Укладач М. І. Безменов

Рецензент Л. М. Любчик

Кафедра системного аналізу і управління

ВСТУП

У багатьох програмах виникає необхідність в уведенні даних за допомогою клавіатури. У Delphi є ряд засобів, призначених для виконання цих дій.

Метою даної самостійної роботи є своєння найпростіших методів уведення даних з клавіатури та їх виведення на екран у програмах, написаних у візуальному середовищі Delphi.

1. ТЕОРЕТИЧНІ ОСНОВИ

Як при клавіатурному введенні, так і при виведенні на екран у візуальному середовищі Delphi використовуються, насамперед, різні редактори. У найпростіших випадках – це однорядковий редактор `TEdit` (або його розвиток – однорядковий редактор `TLabeledEdit`) і багаторядковий редактор `TMemo`. При цьому інформація уводиться й виводиться у рядковому поданні, у зв'язку із чим при введенні або виведенні числових даних необхідно здійснювати відповідне перетворення – від числового подання до рядкового при виведенні або від рядкового подання до необхідного числового формату при введенні. Для забезпечення хоча б найпростішого сервісу необхідно супроводжувати введення даних текстовими підказками, для чого частіше за все використовуть компонент `TLabel` (мітка) з його властивістю `Caption`.

Найпростіша методика введення така:

1. Елемент даних, що вводиться (число, символ, текстовий рядок), вводиться з клавіатури з відображенням його в компоненті `TEdit` (при цьому значення у вигляді рядка записується у властивість `Text`).
2. За деякою командою (наприклад, за подією OnClick, яка генерується кліком мишкою над компонентом `TButton` – кнопка або `TBitBtn` – кнопка з зображенням) проводиться опрацювання введеного рядка. При цьому треба пам’ятати, що властивість `Text` компонента `TEdit` є текстовим рядком, у зв’язку з чим при введенні числових даних уведене значення повинне бути перетворене до числового формату (наприклад, за допомогою функції `StrToInt` при введенні цілочислових даних і `StrToFloat` при введенні дійсних чисел або за допомогою інших засобів).
3. Якщо необхідно організувати повторне введення за допомогою `TEdit` без активізації його за допомогою мишки, то слід передати цьому компоненту так званий фокус уведення, що забезпечується звертанням до методу `SetFocus` цього компонента.

Відзначимо також наступне. У поле введення (рядку редактора) можна занести відразу декілька значень (наприклад, чисел), розділивши їх одним або декількома пробілами. У цьому випадку потрібно подбати про виділення кожного з цих значень.

Нагадаємо також, що в русифікованій версії Windows як роздільник цілої і дробової частин при записуванні дійсних чисел вживається кома, а не десяткова точка, як це має місце в англомовній версії Windows. Щоб уникнути неоднозначності в трактуванні запису дійсних чисел, можна рекомендувати присвоювати системній змінній `DecimalSeparator`, у якій зберігається поділяючий символ, загальноприйнятий у програмуванні поділяючий символ «крапка» (тобто виконати оператор `DecimalSeparator:='.';`).

Однорядковий редактор `TEdit` можна використовувати не тільки для введення, але й для виведення, за необхідності перетворюючи дані, які підлягають виведенню, до рядкового подання за допомогою функцій `IntToStr`, `FloatToStr`, `BoolToStr`, `DateTimeToStr` або за допомогою інших підпрограм. У цьому випадку достатньо виконати присвоювання виведеного рядка властивості `Text` однорядкового редактора.

Для виведення даних можна також скористатися присвоюванням текстового рядка, що виводиься, властивості `Caption` мітки `TLabel`.

У найпростішому випадку для уведення може бути використано кілька однорядкових редакторів (за кількістю даних, що вводяться) і одна кнопка для подачі команди на уведення. У цьому випадку проблем з організацією введення практично немає Головна проблема в цьому випадку – заповнити всі вікна редакторів до подачі команди на уведення. Дещо складніше реалізується методика, орієнтована на використання одного однорядкового редактора з декількома кнопками (по числу даних, що вводяться), які служать для подачі команд на уведення. Все-таки обидва ці підходи не можуть бути використані, коли кількість даних, що вводяться, завелика, – занадто багато компонент потрібно помістити на форму, якщо кількість даних, що вводяться, є досить великою.

Для уведення великої кількості даних можливе використання й одного редактору з однією кнопкою керування уведенням. У цьому випадку необхідна організація підрахунку введених значень зі зміною імен змінних, у які здійснюється запис. Зовсім нескладно організувати при цьому й циклічне введення даних (правда, оператори циклу використати тут не вдасться).

Елементарною є методика организвації уведення послідовності однотипних даних з використанням компонента `TMemo`, що придатний і для виведення, і для введення даних, як і компонент `TEdit`. При введенні даних сформований виведений рядок тексту записується у властивість `Lines` компонента `TMemo` за

допомогою методу Add. Оскільки властивість Lines є доступною і для читання, дані можуть бути набрані у вікні багаторядкового редактора, після чого переписані в змінні, що використовуються в програмі (з можливими перетвореннями, аналогічними тим, що виконуються при введенні за допомогою однорядкового редактора TEdit). Слід ураховувати, що рядки властивості Lines індексуються (нумеруються) від нуля, і при читанні даних звичайно організують цикл (кількість рядків міститься у властивості Lines.Count. Перевагою такого методу введення є те, що до того, як подати команду на уведення, користувач може здійснити перевірку даних, що уводяться, й їх корекцію у вікні редактора.

Досить зручним для організації введення є використання описаної в модулі Dialogs функції InputBox, що служить для виклику діалогового вікна введення, яке дозволяє користувачеві редагувати рядок, що вводиться. При звертанні до функції InputBox на екран виводиться діалогове віконце, аналогічне тому зображеному на рис. 1.1.

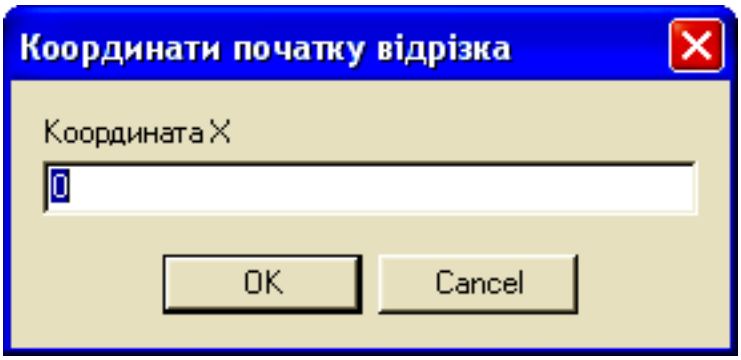

Рис. 1.1.

Віконце містить однорядкове редаговане текстове поле, яке автоматично одержує фокус введення і містить деяке значення за умовчанням. Користувач може або погодитися з відображеним значенням за умовчанням, або набрати в редакторському вікні нове значення. Безпосередньо введення здійснюється кліком мишкою над кнопкою OK (відповідає натисканню клавіші Enter) або кліком мишкою над кнопкою Cancel (відповідає натисканню клавіші Esc).

Функція InputBox має такий формат:

InputBox(заголовок, підказка, рядок_за_умовчуванням).

Тут заголовок – рядок, що задає заголовок (на рис. 3.1 – Координати початку відрізка);

підказка – рядок-підказка для користувача (на рис. 3.1 – Координата x);

рядок_за_умовчуванням – рядок, що з'являється в однорядковому редакторі даного віконця при його виведенні на екран і вводиться при натисканні кнопки Cancel (на рис. 1.1 – рядок 0) .

При виборі користувачем кнопки Cancel (або натисканні клавіші Esc) функція повертає значення за умовчанням (третій параметр), при виборі ж кнопки OK (або натисканні клавіші Enter) функція повертає значення, що міститься в однорядковому редакторі.

Функція повертає рядок, який за необхідності може бути перетворений до потрібного формату за допомогою однієї з функцій `StrToXXX`.

Функцію `InputBox` рекомендується використовувати в тому випадку, коли додатку байдуже, яка кнопка вибиралася – OK або Cancel. Якщо ж додаток повинен розпізнавати обрану кнопку, рекомендується застосовувати функцію `InputQuery`.

Функція `InputQuery` має ті ж самі три параметри, що й функція `InputBox`, але третій параметр є таким, за допомогою якого повертається або набране в редакторському вікні значення (якщо вибиралася кнопка OK), або значення за умовчанням (при виборі кнопки Cancel). Записане в третій параметр значення і є введеним значенням. Результат роботи функції `InputQuery` є – значення: `True` при виборі кнопки OK або `False` при виборі кнопки Cancel.

Особливістю введення за допомогою функцій `InputBox` і `InputQuery` є те, що звертання до цих функцій може бути здійснене багаторазово й не сполучене з виконанням яких-небудь опрацьовувачів подій. Ці функції зручно використовувати при циклічному введенні, зокрема, при введенні масивів.

2. ПРИКЛАД ПРОГРАМИ

2.1. Одна кнопка та декілька редакторів

Квадратне рівняння $ax^2 + by + c = 0$ задане своїми коефіцієнтами. Обчислити дискримінант цього рівняння.

Розв'язання. Створимо форму, помістивши на неї панель TPanel та багаторядковий редактор TMemo, після чого змінимо деякі з їхніх властивостей.

Форма:

Caption — Уведення даних

Height — 390

Width — 690

Панель:

Caption — очистимо

Height — 271

Align — alBottom

Багаторядковий редактор:

Name — mmOutput1

Align — alCustom

Lines — очистимо

Помістимо на панель три однорядкових редактори TEdit, над кожним з яких мітки TLabel. Крім того, розмістимо на панелі дві звичайні кнопки TButton і графічну кнопку TBitBtn. Змінимо деякі властивості цих компонентів.

Перший однорядковий редактор:

Name — edInput1

Text — 1

Перша мітка:

Name — lbOutput11

Caption — Коефіцієнт a

Visible — False

Другий однорядковий редактор:

Name — edInput2

Text — очистимо

Visible — False

Друга мітка:

Name — lbOutput12

Caption — Коефіцієнт b

Visible — False

Третій однорядковий редактор:

Name — edInput3

Text — очистимо

Visible — False

Третя мітка:

Name — lbOutput13

Caption — Коефіцієнт c

Visible — False

Перша кнопка (для команди на введення даних):

Name — btInput1

Caption — Увести

Друга кнопка (для команди на обчислення):

Name — btRun1

Caption — Обчислити

Графічна кнопка:

Kind — bkClose

Змінюючи за допомогою мишки положення деяких компонентів досягнемо вигляду форми, аналогічному тому, що зображений на рис 2.1

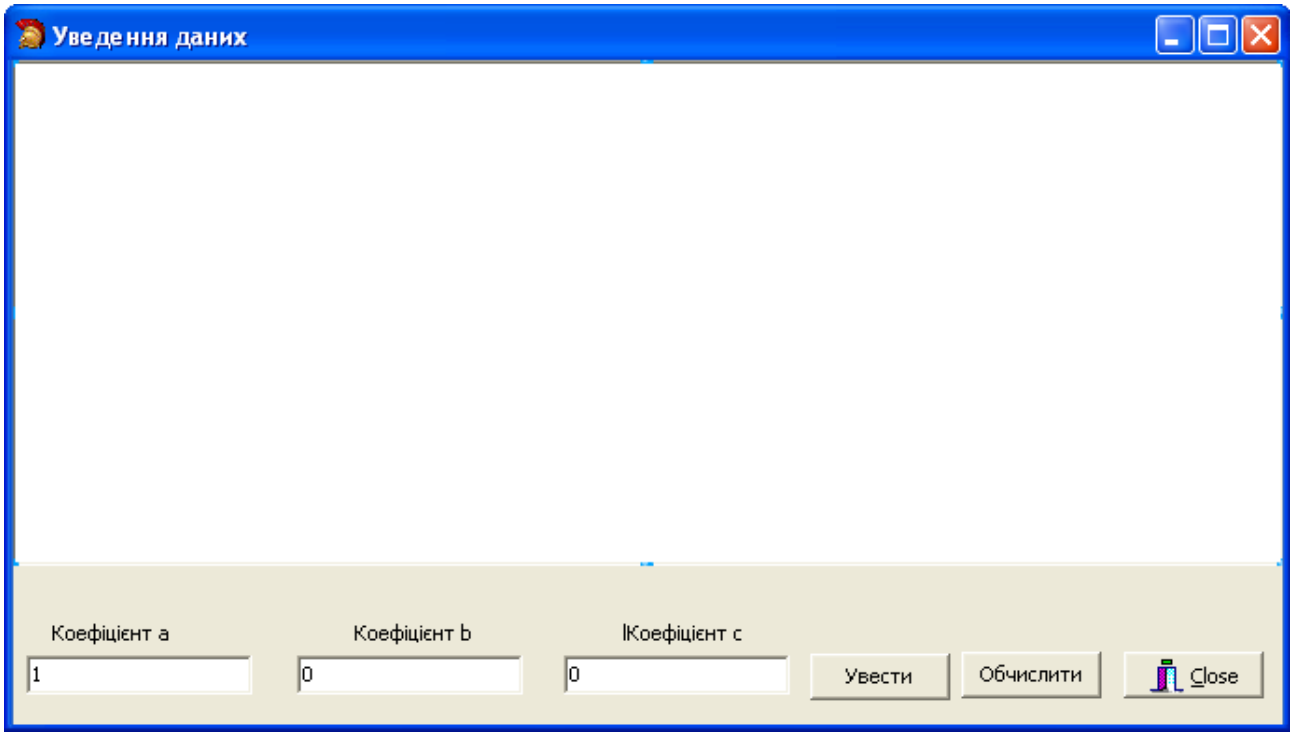

Рис. 2.1.

У секції **private** (або **public**) опису форми опишемо також такі поля:

```
a, b, c: Real;
```

Для розв’язання задачі використуємо такі опрацьовувачі подій:

```
procedure TForm1.btInput1Click(Sender: TObject);
begin
  a := StrToInt(edInput1.Text);
  b := StrToInt(edInput2.Text);
  c := StrToInt(edInput3.Text);
  mmOutput1.Lines.Add('a = ' + FloatToStr(a));
  mmOutput1.Lines.Add('b = ' + FloatToStr(b));
  mmOutput1.Lines.Add('c = ' + FloatToStr(c));
  edInput1.Visible := False;
  lbOutput1.Visible := False;
  edInput2.Visible := False;
  lbOutput2.Visible := False;
```

```
  edInput3.Visible := False;
  lbOutput3.Visible := False;
  btInput1.Visible := False;
  btRun1.Visible := True;
end;

procedure TForm1.btRun1Click(Sender: TObject);
var
  d: Real;
begin
  d := b * b - 4 * a * c;
  mmOutput1.Lines.Add('Дискримінант дорівнює ' +
                      FloatToStr(d));
  btRun1.Visible := False;
end;

procedure TForm1.FormCreate(Sender: TObject);
begin
  DecimalSeparator := '.';
  edInput1.TabOrder := 0;
end;
```

2.2. Один редактор та декілька кнопок

Скористаємося формою з одним однорядковим редактором і трьома кнопками для розв'язання задачі з п. 2.1.

Розв'язання. Внесемо деякі зміни у перелік компонентів форми, що була розроблена у п. 2.1, а саме, вилучимо редактори edInput2 та edInput3 і мітки lbOutput12 та lbOutput13 і додамо ще дві звичайні кнопки, надавши їм імена btInput2 та btInput3. За допомогою мишки «перетягнемо» кнопки btInput1, btInput2, btInput3, btRun1 ближче до однорядкового редактора, наклавши їх одну на одну так, щоб створювалося враження наявності тільки однієї кнопки. «Підтягнемо» також і графічну кнопку. Надамо також властивості Visible кнопок btInput2, btInput3, btRun1 значення False. Крім того, запишемо у властивість Caption кнопок уведення такі написи:

для btInput1 — Уведення a
для btInput2 — Уведення b
для btInput3 — Уведення c

Змінимо також розміри форми, досягнувши в результаті її вигляду, зображеного на рис. 1.2.

Нижче наведено можливі тексти опрацьовувачів події OnClick для кнопок та події OnCreate форми.

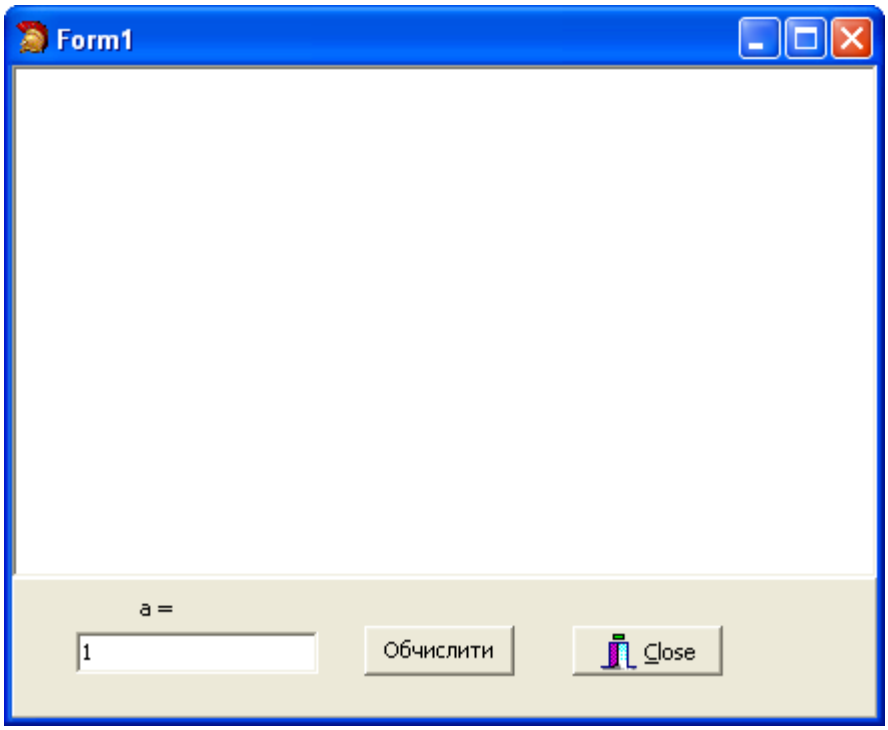

Рис. 2.1.

```
procedure TForm1.btInput1Click(Sender: TObject);
begin
  a := StrToInt(edInput1.Text);
  mmOutput1.Lines.Add('a = ' + FloatToStr(a));
  lbOutput1.Caption := 'b =';
  btInput1.Visible := False;
  btInput2.Visible := True;
  edInput1.SetFocus;
end;

procedure TForm1.btInput2Click(Sender: TObject);
begin
  b := StrToInt(edInput1.Text);
  mmOutput1.Lines.Add('b = ' + FloatToStr(b));
  lbOutput1.Caption := 'c =';
  btInput2.Visible := False;
  btInput3.Visible := True;
  edInput1.SetFocus;
```

```
end;

procedure TForm1.btInput3Click(Sender: TObject);
begin
  c := StrToInt(edInput1.Text);
  mmOutput1.Lines.Add('c = ' + FloatToStr(c));
  lbOutput1.Visible := False;
  edInput1.Visible := False;
  btInput3.Visible := False;
  btRun1.Visible := True;
end;

procedure TForm1.btRun1Click(Sender: TObject);
var
  d: Real;
begin
  d := b * b - 4 * a * c;
  mmOutput1.Lines.Add('Дискримінант дорівнює ' +
                      FloatToStr(d));
  btRun1.Visible := False;
end;

procedure TForm1.FormCreate(Sender: TObject);
begin
  DecimalSeparator := '.';
  edInput1.TabOrder := 0;
end;
```

2.3. Один редактор та одна кнопка

Розв’язується та ж задача.

Розв’язання. Скористаємося формою з п. 2.2 (див. рис. 2.2), видаливши з неї кнопки btInput2 та btInput3 (у властивості Visible кнопки, btRun1 залишимо значення False). Можна скористатися такими опрацьовувачами подій (властивість Tag типу `Integer` визначена у всіх компонентіві за умовчанням має значення 0):

```
procedure TForm1.btInput1Click(Sender: TObject);
begin
  edInput1.SetFocus;
```

```
  case Tag of
    0: begin
         a := StrToInt(edInput1.Text);
         mmOutput1.Lines.Add('a = ' + FloatToStr(a));
         lbOutput1.Caption := 'b =';
       end;
    1: begin
         b := StrToInt(edInput1.Text);
         mmOutput1.Lines.Add('b = ' + FloatToStr(b));
         lbOutput1.Caption := 'c =';
       end;
    2: begin
         c := StrToInt(edInput1.Text);
         mmOutput1.Lines.Add('c = ' + FloatToStr(c));
         lbOutput1.Visible := False;
         edInput1.Visible := False;
         btInput1.Visible := False;
         btRun1.Visible := True;
       end;
  end;
  Tag := Tag + 1;
end;

procedure TForm1.btRun1Click(Sender: TObject);
var
  d: Real;
begin
  d := b * b - 4 * a * c;
  mmOutput1.Lines.Add('Дискримінант дорівнює ' +
                      FloatToStr(d));
  btRun1.Visible := False;
end;

procedure TForm1.FormCreate(Sender: TObject);
begin
  DecimalSeparator := '.';
  edInput1.TabOrder := 0;
end;
```

2.4. Циклічне введення за допомогою одного редактора та однієї кнопки

Уводяться дійсні числа до першого від'ємного, яке є ознакою закінчення введкння. Обчислити суму чисел, що були введені.

Розв'язання. Скористаємося формою з п. 2.3, видаливши з неї кнопку btRun1 та записавши у властивість Caption мітки текст Уведіть число, а у властивість Text однорядкового редактора число 0. У секції **public** опису форми визначимо поле Sum типу Real (якщо форма не створюється заново, а модифікується, – видалити опис поліва a, b, c) і скористаємося такими опрацьовувачами подій:

```
procedure TForm1.btInput1Click(Sender: TObject);
var
  a: Real;
begin
  edInput1.SetFocus;
  a := StrToFloat(edInput1.Text);
  if a >= 0 then Sum := Sum + a
  else begin
    mmOutput1.Lines.Add(FloatToStr(Sum));
    btInput1.Visible := False;
    edInput1.Visible := False;
    lbOutput1.Visible := False;
  end;
end;

procedure TForm1.FormCreate(Sender: TObject);
begin
  DecimalSeparator := '.';
  edInput1.TabOrder := 0;
  Sum := 0;
end;
```

2.5. Використання багаторядкового редактора для циклічного введення

Дано послідовність дійсних чисел. Обчислити їх суму.

Розв'язання. Скористаємося формою з п. 2.3, видаливши з неї однорядковий редактор і мітку і записавши у властивість Caption кнопки btInput1 текст Обчислити. У секції **public** опису форми визначимо поле `Sum` типу `Real` (якщо форма не створюється заново, а модифікується, – видалити опис полів `a`, `b`, `c`) і скористаємося такими опрацьовувачами подій:

```
procedure TForm1.btRun1Click(Sender: TObject);
var
  i: Integer;
begin
  Sum := 0;
  for i := 0 to mmInpOut1.Lines.Count - 1 do
    Sum := Sum + StrToFloat(mmInpOut1.Lines[i]);
  mmInpOut1.Lines.Add('Sum = ' + FloatToStr(Sum));
  btRun1.Visible := False;
end;

procedure TForm1.FormCreate(Sender: TObject);
begin
  DecimalSeparator := '.';
  mmInpOut1.TabOrder := 0;
end;
```

2.6. Використання функції InputBox

Дано послідовність дійсних чисел. Обчислити їх суму.

Розв'язання. Скористаємося формою з п. 2.4, видаливши з неї однорядковий редактор і мітку (нижче змінна `Sum` визначена усередині процедури `TForm1.btRun1Click`, а не в описі форми:

```
procedure TForm1.btRun1Click(Sender: TObject);
var
  i: Integer;
  a, Sum: Real;
begin
  Sum := 0;
  a := StrToFloat(InputBox(
           'Уведення невід''ємних чисел',
           'Ознака кінця уведення - від''ємне число','0'));
  while a >= 0 do begin
```

```
    Sum := Sum + a;
    mmInpOut1.Lines.Add(FloatToStr(a));
    a := StrToFloat(InputBox(
           'Уведення невід''ємних чисел',
           'Ознака кінця уведення - від''ємне число','0'));
  end;
  mmInpOut1.Lines.Add('Sum = ' + FloatToStr(Sum));
  btRun1.Visible := False;
end;
procedure TForm1.FormCreate(Sender: TObject);
begin
  DecimalSeparator := '.';
end;
```

СПИСОК ЛІТЕРАТУРИ

1. Безменов, М. І. Основи програмування в середовищі Delphi : навч. посіб. / М. І. Безменов. – Х. : НТУ «ХПІ», 2010. – 608 с.
2. Кэнту, М. Delphi 7 : Для профессионалов / М. Кэнту – СПб. : Питер, 2004. – 1101 с.
3. Архангельский, А. Я. Программирование в Delphi 6 / А. Я. Архангельский. – М. : БИНОМ, 2002. – 1120 с.
4. Дарахвелидзе, П. Г. Программирование в Delphi 7 / П. Г. Дарахвелидзе, Е. П. Марков. – СПб. : БХВ-Петербург, 2003. – 784 с.
5. Культин, Н. Б. Основы программирования в Delphi 7 / Н. Б. Культин. – СПб. : БХВ-Петербург, 2003. – 608 с.
6. Пестриков, В. М. Delphi на примерах / В. М. Пестриков, А. Н. Маслобоев. – СПб. : БХВ-Петербург, 2005. – 496 с.
7. Ремкеев, А. А. Курс Delphi для начинающих. Полигон нестандартных задач / А. А. Ремкеев, С. В. Федотова. – М. : СОЛОН-Пресс, 2006. – 360 с.
8. Митчелл, К. Керман. Программирование и отладка в Delphi : учебный курс / Митчелл К. Керман. – М. : Вильямс, 2004. – 720 с.
9. Парижский, С. М. Delphi : Только практика / С. М. Парижский. – К. : МК-Пресс, 2005. – 208 с.
10. Культин, Н. Б. Основы программирования в Delphi 2007 / Н. Б. Культин. – СПб. : БХВ-Петербург, 2008. – 480 с.

Навчальне видання

Методичні вказівки
до самостійної роботи
«Найпростіші введення та виведення у візуальному середовищі Delphi»
з курсу «Програмування» для студентів напряму 6.040302 – Інформатика
(спеціалізація «Соціальна інформатика»)

Укладач БЕЗМЕНОВ Микола Іванович

Відповідальний за випуск О. С. Куценко
Роботу до видання рекомендував О. В. Горелий

За авторською редакцією

План 2011 р., поз. 11 / 69-11

Підп. до друку 23.05.2011 р. Формат 60 × 84 $^{1}/_{16}$. Папір офісний.
Riso-друк. Гарнітура Таймс. Ум. друк. арк. 0,9. Наклад 50 прим.
Зам. № 166. Ціна договірна.

Видавничий центр НТУ «ХПІ».
Свідоцтво про державну реєстрацію ДК № 3657 від 24.12.2009 р.
61002, Харків, вул. Фрунзе, 21

Друкарня НТУ «ХПІ», 61002, Харків, вул. Фрунзе, 21